Safety-aware Adaptive Reinforcement Learning with Applications to Brushbot Navigation

Motoya Ohnishi, *Student Member, IEEE*, Li Wang, *Student Member, IEEE*,
Gennaro Notomista, *Student Member, IEEE*, and Magnus Egerstedt, *Fellow, IEEE*

***Abstract*— This paper presents a safety-aware learning framework that employs an adaptive model learning method together with barrier certificates for systems with possibly nonstationary agent dynamics. To extract the dynamic structure of the model, we use a sparse optimization technique, and the resulting model will be used in combination with control barrier certificates which constrain feedback controllers only when safety is about to be violated. Under some mild assumptions, solutions to the constrained feedback-controller optimization are guaranteed to be globally optimal, and the monotonic improvement of a feedback controller is thus ensured. In addition, we reformulate the (action-)value function approximation to make any kernel-based nonlinear function estimation method applicable. We then employ a state-of-the-art kernel adaptive filtering technique for the (action-)value function approximation. The resulting framework is verified experimentally on a *brushbot*, whose dynamics is unknown and highly complex.**

***Index Terms*— Safe learning, control barrier certificate, sparse optimization, kernel adaptive filtering, brushbot**

I. Introduction

By exploring and interacting with an environment, reinforcement learning can successfully determine the optimal feedback controller with respect to the long-term rewards received by an agent [1], [2]. Whereas the idea of determining the optimal feedback controller in terms of a cost over some time horizon is standard in the controls literature [3], reinforcement learning is aimed at learning the long-term rewards by exploring the states and actions. As such, the agent dynamics is no longer explicitly taken into account, but rather is subsumed by the data. Moreover, even the rewards need not necessarily be known a priori, but can be obtained through exploration, as well.

If no information about the agent dynamics is available, however, an agent might end up in certain regions of the state space which must be avoided while exploring. Avoiding such regions of the state space is referred to as *safety*. Safety includes collision avoidance, boundary-transgression avoidance, connectivity maintenance in teams of mobile robots, and other mandatory constraints, and this tension between exploration and safety becomes particularly pronounced in robotics, where safety is crucial.

In this paper, we address this safety issue, by employing model learning in combination with barrier certificates. In particular, we focus on learning for a system with discrete-time nonstationary agent dynamics. Nonstationarity comes, for example, from failures of actuators, battery degradations, or sudden environmental disturbances. The result is a method that adapts to a time-varying agent dynamics and simultaneously extracts the dynamic structure without having to know how the agent dynamics changes over time. The resulting model will be used for barrier certificates. Moreover, certain conditions are also presented under which the monotonic improvement of a barrier-certified feedback controller is ensured.

Over the last decade, the safety issue has been addressed under the name of safe learning, and plenty of solutions have been proposed [4]–[13]. To ensure safety while exploring, an initial knowledge of the agent dynamics, some safe maneuver or their long-term rewards, or a *teacher* advising the agent is necessary [4], [14]. To obtain a model of the agent dynamics, human operators may maneuver the agent and record its trajectories [12], [15], or, starting from an initial safe maneuver, the set of safe feedback controllers can be expanded by exploring the states [4], [5]. It is also possible that an agent continues exploring without entering the states with low long-term rewards associated with some safe maneuver (e.g. [16]). Due to the inherent uncertainty, the worst case scenario (e.g. possible lowest rewards) is typically taken into account when expanding the set of safe feedback controllers [13], [17]. To address the issue of this uncertainty for nonlinear-model estimation tasks, Gaussian process regression [18] is a strong tool, and many safe learning studies have taken advantage of its property (e.g. [4], [6], [7], [10], [13]).

Nevertheless, when the agent dynamics is nonstationary, the assumptions often made in the safe learning literature cannot hold any more. In such cases, we need to adaptively learn the agent dynamics to mitigate the effect of an unexpected violation of safety. Moreover, when the agent dynamics varies, the long-term rewards associated with states and controls also vary. As such, the long-term rewards must also be learned in an adaptive manner. These are the core motivations of this paper.

To constrain the states within a desired safe region, we employ control barrier functions (cf. [19]–[24]). When the accurate model of the agent dynamics is available, control barrier certificates ensure that an agent remains in the safe states for all time by constraining the instantaneous control

This work of M. Ohnishi was supported by Scandinavia-Japan Sasakawa Foundation.

M. Ohnishi is with the School of Electrical Engineering, KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: motoya@kth.se). L. Wang and M. Egerstedt are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA (e-mail: liwang@gatech.edu; magnus@gatech.edu). G. Notomista is with the School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA (e-mail: g.notomista@gatech.edu).

input at each time, and that an agent outside of the safe states is forced back to safety. A useful property of control barrier certificates is its non-conservativeness, i.e., it modifies feedback controllers when violations of safety are truly imminent. On the other hand, the global optimality of solutions to the constrained feedback-controller optimization is necessary to ensure the monotonic improvement of a feedback controller. Our first contribution of this paper is to propose a discrete-time control barrier certificate which ensures the global optimality under some mild conditions. This is an improvement of the previously proposed discrete-time control barrier certificate [24], and hence has wider applicability.

To adaptively learn a nonlinear model of the agent dynamics, we employ kernel adaptive filtering [25], which is an adaptive extension of the kernel ridge regression [26], [27] or Gaussian processes. Multikernel adaptive filtering (cf. [28]–[32]) is a state-of-the-art kernel adaptive filtering, which adaptively achieves a compact representation of a nonlinear function containing multi-component/partially-linear functions, and has a monotone approximation property for a possibly varying target function. Our second contribution of this paper is to regard a model of the agent dynamics as a combination of multiple structural components, and apply multikernel adaptive filtering to simultaneously learn a model and the dynamic structure. The key idea is the use of an adaptive sparse optimization to extract truly *active* structural components.

Lastly, the (action-)value function, which approximates the long-term rewards, needs to be adaptively estimated under nonstationarity. Therefore, we wish to fully exploit the nonlinear adaptive filtering techniques. Actually, many attempts have been made to apply the online learning techniques to reinforcement learning (see [33]–[38]). As a result, so-called off-policy approaches, which are convergent even when samples are *not* generated by the target feedback controller (see [34]), have been proposed. However, what differentiates the (action-)value function approximation from an ordinary supervised learning, where input-output pairs are given, is that the output of the true (action-)value function is not explicitly observed. Our final contribution of this paper is, by assuming deterministic agent dynamics, to appropriately reformulate the (action-)value function approximation problem so that any kernel-based learning, which is widely-studied nonparametric technique, becomes straightforwardly applicable.

To test the efficacy, the proposed learning framework is implemented on a *brushbot*, whose dynamics is unknown and highly complex, and we conduct an experiment in the real world. This is challenging due to many uncertainties and lack of simulators often used in applications of reinforcement learning to robotics (see [39] for reinforcement learning in robotics).

II. Background Material

We introduce notation, related works on safe learning, and background material, including control barrier functions, and kernel adaptive filtering.

A. Notation

Throughout, $\mathbb{R}$, $\mathbb{N}$, and $\mathbb{N}^*$ are the sets of real numbers, nonnegative integers, and positive integers, respectively. Given any pair of integers $m, n \in \mathbb{N}$ such that $m \leq n$, we denote the set $\{m, m+1, \cdots, n\}$ by $\overline{m,n}$.

The state and the control input at time instance $n \in \mathbb{N}$ are represented by $x_n \in \mathcal{X} \subset \mathbb{R}^{n_x}$, and $u_n \in \mathcal{U} \subset \mathbb{R}^{n_u}$, where $n_x, n_u \in \mathbb{N}^*$, respectively. The functions $R : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$, $V^\phi : \mathcal{X} \to \mathbb{R}$, and $Q^\phi : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ represent the instantaneous-reward function, the value function, and the action-value function associated with a feedback controller $\phi : \mathcal{X} \to \mathcal{U}$, respectively. The estimate of a function φ at time instance n is denoted by $\varphi^{(n)}$. We denote the discount factor for the (action-)value function approximation by $\gamma \in (0, 1)$.

The metric projection of a point $x \in \mathbb{R}^L$ onto a given closed convex set $C \subset \mathbb{R}^L$ is defined by $P_C(x) := \operatorname{argmin}_{y \in C} \|x - y\|_{\mathbb{R}^L}$. Let $\|\cdot\|_{\mathcal{H}}$ be the norm induced by the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ in an inner-product space $\mathcal{H}$. In particular, define $\langle x, y \rangle_{\mathbb{R}^L} := x^\mathsf{T} y$ for L-dimensional real vectors $x, y \in \mathbb{R}^L$, and $\|x\|_{\mathbb{R}^L} := \sqrt{\langle x, x \rangle_{\mathbb{R}^L}}$, where $(\cdot)^\mathsf{T}$ stands for transposition. The vector e_i denotes the unit vector with one at the ith position and zeros elsewhere. We denote the null (zero) function by 0, and the empty set by $\emptyset$.

B. Related Work on Safe Learning

The primary focus of this paper is the safety issue *while exploring*. Typically, some initial knowledges, such as an initial safe controller and a model of the agent dynamics, are required to address the safety issue while exploring, and model learning is often employed together. In this section, we introduce some related work on safe learning with model learning, and present a brief summary of existing (action-)value function approximation techniques.

1) Model Learning for Safe Maneuver: The recent work in [13], [7], and [4] assume an initial conservative set of safe controllers, which is gradually expanded as more data become available. These approaches are designed for stationary agent dynamics, and Gaussian processes are employed to obtain the confidence interval of the model. To ensure safety, control barrier functions and control Lyapunov functions are employed in [13] and [4], respectively. On the other hand, the work in [10] uses a trajectory optimization based on the receding horizon control and model learning by Gaussian processes, which is computationally expensive when the model is highly nonlinear.

In contrast to these approaches, our approach considers a possibly nonstationary agent dynamics and employs adaptive model learning to adaptively estimate the time-varying set of safe controllers.

2) (Action-)value Function Approximation: We introduce, briefly, ideas of existing (action-)value function approximation techniques. The Bellman equation defines the temporal difference error:

$$V^{\phi^{(n)}}(x_n) - T^\phi(V^{\phi^{(n)}}(x_n)), \tag{1}$$

where $T^\phi(V^{\phi^{(n)}}(x_n)) := \gamma V^{\phi^{(n)}}(x_{n+1}) + R(x_n, \phi(x_n))$, at time instance $n+1$. Since the outputs of the value function

is not directly observed, supervised learning methods cannot be directly applied. The update rule of the temporal difference learning [40], [41] is given by

$$V^{\phi^{(n+1)}}(x_n) = \lambda \left[T^{\phi}(V^{\phi^{(n)}}(x_n)) - V^{\phi^{(n)}}(x_n) \right],$$

where λ is the step size. Note that the output depends on the current estimator of the value function in this update rule. On the other hand, so-called off-policy methods (e.g. the residual learning [33], the least squares temporal difference algorithm [42], and the gradient temporal difference learning [34], [43]) have been also proposed. These approaches are proved to converge under certain conditions as like most supervised learning methods even when samples are not generated by the controller ϕ. The least squares temporal difference algorithm has been extended to kernel-based methods [37], including Gaussian processes (e.g. Gaussian process temporal difference and Gaussian process SARSA [35]).

Unlike these kernel-based methods, our approach explicitly defines a so-called reproducing kernel Hilbert space so that the (action-)value function approximation becomes an ordinary kernel-based supervised learning, and that any kernel-based method can be straightforwardly applied without modifying.

We refer the readers to [44] for a summary of parametric value function approximation techniques.

C. Discrete-time Control Barrier Function

To avoid certain regions of the state space, we employ control barrier functions, which only modify feedback controllers when safety is about to be violated. Define the set of safe states $\mathcal{C} \subset \mathcal{X}$ as

$$\mathcal{C} := \{x \in \mathcal{X} | B(x) \geq 0\}, \tag{2}$$

where $B : \mathcal{X} \to \mathbb{R}$ is called the discrete-time exponential barrier function.

Definition 1 ([24, Definition 4]). *A map $B : \mathcal{X} \to \mathbb{R}$ is a discrete-time exponential control barrier function if there exists a control input $u_n \in \mathcal{U}$ such that*

$$B(x_{n+1}) - B(x_n) \geq -\eta B(x_n), \ \forall n \in \mathbb{N}, \ 0 < \eta \leq 1. \tag{3}$$

Note that we intentionally removed the condition $B(x_0) \geq 0$ originally presented in [24, Definition 4]. Then, the forward invariance and asymptotic stability of the set of safe states are ensured by the following proposition.

Proposition 1. *The set $\mathcal{C}$ defined in (2) for some discrete-time exponential control barrier function $B : \mathcal{X} \to \mathbb{R}$ is forward invariant when $B(x_0) \geq 0$, and is asymptotically stable when $B(x_0) < 0$.*

Proof. See [24, Proposition 4] for the proof of forward invariance. The set $\mathcal{C} \subset \mathcal{X}$ is asymptotically stable as

$$\lim_{n\to\infty} B(x_n) \geq \lim_{n\to\infty} (1-\eta)^n B(x_0) = 0,$$

where the inequality holds from [24, Proposition 1]. □

Proposition 1 implies that an agent remains in the set of safe states defined in (2) for all time if $B(x_0) \geq 0$ and (3) are satisfied, and the agent outside of the set of safe states is forced back to safety.

D. Multikernel Adaptive Filtering–Use of Sparse Optimization

As we will see later in Section III-A, barrier certified controllers are efficiently computed when the agent dynamics is affine to control inputs. Here, we introduce the idea of multikernel adaptive filtering, which will be used to simultaneously learn a model and the dynamic structure. We emphasize that one advantage of kernel-based methods is their convex-analytic formulations.

Kernel adaptive filtering is a tool of nonlinear function estimation. Due to its celebrated property of reproducing kernels, the framework of linear adaptive filtering is directly applied to nonlinear function estimation tasks in a possibly infinite-dimensional functional space, namely a reproducing kernel Hilbert space.

Definition 2 ([45, page 343]). *Given a nonempty set $\mathcal{Z}$ and $\mathcal{H}$ which is a Hilbert space defined in $\mathcal{Z}$, the function $\kappa(z, w)$ of $z, w \in \mathcal{Z}$ is called a reproducing kernel of $\mathcal{H}$ if*

1) *for every $w \in \mathcal{Z}$, $\kappa(z, w)$ as a function of $z \in \mathcal{Z}$ belongs to $\mathcal{H}$, and*
2) *it has the reproducing property, i.e., the following holds for every $w \in \mathcal{Z}$ and every $\varphi \in \mathcal{H}$ that*

$$\varphi(w) = \langle \varphi, \kappa(\cdot, w) \rangle_{\mathcal{H}}.$$

If $\mathcal{H}$ has a reproducing kernel, $\mathcal{H}$ is called a Reproducing Kernel Hilbert Space (RKHS).

One of the celebrated examples of kernels is the Gaussian kernel $\kappa(z, w) := \frac{1}{(2\pi\sigma^2)^{L/2}} \exp\left(-\frac{\|z-w\|^2_{\mathbb{R}^L}}{2\sigma^2}\right)$, $z, w \in \mathbb{R}^L$, $\sigma > 0$. It is well-known that the Gaussian reproducing kernel Hilbert space has universality [46], i.e, any continuous function on every compact subset of $\mathbb{R}^L$ can be approximated with an arbitrary accuracy. Another widely used kernel is the polynomial kernel $\kappa(z, w) := (z^{\mathsf{T}} w + c)^d$, $c \geq 0, d \in \mathbb{N}^*$.

Multikernel adaptive filtering [28] exploits multiple kernels to conduct learning in the sum space of RKHSs associated with each kernel. Let $M \in \mathbb{N}^*$ be the number of kernels employed. Denote, by $\mathcal{D}_{m,n} := \{\kappa_m(\cdot, \tilde{z}_{m,j})\}_{j \in \overline{1, r_{m,n}}}$, $m \in \overline{1, M}, r_{m,n} \in \mathbb{N}^*$, the time-dependent set of functions, referred to as a *dictionary*, at time instance n for the mth kernel $\kappa_m(\cdot, \cdot)$. The current estimator $\psi_n^{(n)}$ is evaluated at the current input z_n, in a linear form, as

$$\psi_n^{(n)}(z_n) := h_n^{\mathsf{T}} k_n(z_n) = \sum_{m=1}^{M} h_{m,n}^{\mathsf{T}} k_{m,n}(z_n), \tag{4}$$

where $h_n := [h_{1,n}^{\mathsf{T}}, h_{2,n}^{\mathsf{T}}, \cdots, h_{M,n}^{\mathsf{T}}]^{\mathsf{T}} := [h_1, h_2, \cdots, h_{r_n}] \in \mathbb{R}^{r_n}$, $r_n := \sum_{m \in \overline{1,M}} r_{m,n}$, is the coefficent vector, and $k_n(z_n) := \left[k_{1,n}(z_n)^{\mathsf{T}} k_{2,n}(z_n)^{\mathsf{T}} \cdots k_{M,n}(z_n)^{\mathsf{T}}\right]^{\mathsf{T}} \in \mathbb{R}^{r_n}$, $k_{m,n}(z_n) := \left[\kappa_m(z_n, \tilde{z}_{m,1}), \kappa_m(z_n, \tilde{z}_{m,2}), \cdots, \kappa_m\left(z_n, \tilde{z}_{m,r_{m,n}}\right)\right]^{\mathsf{T}} \in \mathbb{R}^{r_{m,n}}$. An illustrative example of an estimator is given in Figure 1, where ψ_n is the unknown function to be

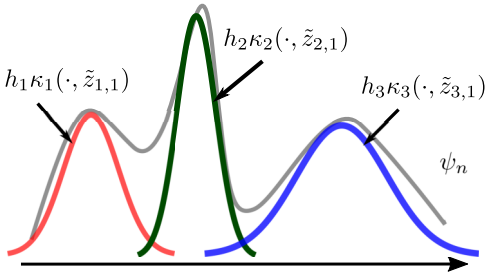

Fig. 1. An illustration of multikernel adaptive filtering. By employing multiple kernel functions, a compact representation of the unknown function ψ_n to be estimated is achieved. Gaussian kernels $\kappa_m, m \in \overline{1,3}$, with different scale parameters (i.e width) are employed in this example.

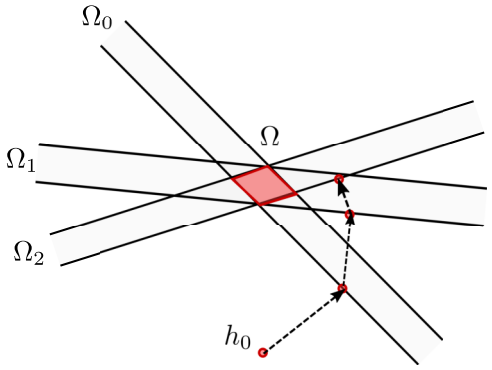

Fig. 2. An illustration of monotone approximation property. The estimate h_n monotonically approaches to the set Ω of optimal vectors h^* by sequentially minimizing the distance between h_n and Ω_n.

estimated at time instance n. To curve the growth of dictionary sizes and obtain a compact representation of the unknown function to be estimated, a sparse optimization can be applied. Given an input-output pair $(z_n, \delta_n), z_n \in \mathbb{R}^L, \delta_n \in \mathbb{R}$, at time instance n, define a closed convex set $C_\ell \subset \mathbb{R}^{r_n},\ \ell \in \overline{n-s_n+1, n} \subset \mathbb{N},\ s_n \in \mathbb{N}^*$ as

$$C_\ell := \{h \in \mathbb{R}^{r_n} | |h^\mathsf{T} k_n(z_\ell) - \delta_\ell| \leq \rho\},\ \rho \geq 0, \tag{5}$$

which is a set of coefficient vector h satisfying instantaneous-error-zero with a precision parameter ρ. The cost for a sparse optimization at time instance n is defined by

$$\Theta_n(h) := \frac{1}{2} \sum_{\ell=n-s_n+1}^{n} \frac{1}{s_n} dist^2(h, C_\ell) + \mu \|h\|_1, \tag{6}$$

where $dist(h, C_\ell) := \min_{a \in C_\ell} \|h - a\|_{\mathbb{R}^{r_n}}$, and the ℓ_1-norm regularization $\|h\|_1 := \sum_{i=1}^{r_n} |h_i|$ with a parameter $\mu \geq 0$ promotes sparsity of h. The update rule of the adaptive proximal forward-backward splitting [47], which is an adaptive filtering designed for sparse optimizations, for the cost (6) is given by

$$h_{n+1} = \text{prox}_{\lambda\mu} \left[(1-\lambda) I + \lambda \sum_{\ell=n-s_n+1}^{n} \frac{1}{s_n} P_{C_\ell} \right] (h_n), \tag{7}$$

where $\lambda \in (0, 2)$ is the step size, I is the identity operator, and

$$\text{prox}_{\lambda\mu}(h) = \sum_{i}^{r_n} \text{sgn}(h_i) \max\{|h_i| - \lambda\mu, 0\} e_i, \tag{8}$$

where $\text{sgn}(\cdot)$ is the sign function. Then, the strictly monotone approximation property [47]: $\|h_{n+1} - h_n^*\|_{\mathbb{R}^{r_n}} < \|h_n - h_n^*\|_{\mathbb{R}^{r_n}},\ \forall h_n^* \in \Omega_n := \text{argmin}_{h \in \mathbb{R}^{r_n}} \Theta_n(h)$, holds if $h_n \notin \Omega_n \neq \emptyset$. Under nonstationarity, this monotone approximation property tells that, no matter how the target vector(function) changes, we can at least guarantee that the current estimator h_n gets closer to the current target vector. This property plays a key role in the safety-aware adaptive reinforcement learning presented in Section III.

Assume that the dictionary is fixed for $n \geq N$ for some $N \in \mathbb{N}$, i.e., $\mathcal{D}_{m,n} = \mathcal{D}_{m,N}, \forall n \geq N,\ m \in \overline{1, M}$, and that $\Omega := \bigcap_{n \geq N} \Omega_n$ is nonempty. Then, $\|h_{n+1} - h^*\|_{\mathbb{R}^{r_N}} < \|h_n - h^*\|_{\mathbb{R}^{r_N}},\ \forall h^* \in \Omega,\ n \geq N$, holds if $h_n \notin \Omega_n$ (see [48] for detail). This is illustrated in Figure 2.

Dictionary Construction: By using sparse optimizations, *nonactive* structural components represented by some kernel functions are pruned, and the dictionary is refined as time goes by. On the other hand, we employ two novelty conditions when adding the kernel functions $\{\kappa_m(\cdot, z_n)\}_{m \in \overline{1,M}}$ to the dictionary: (i) the maximum-dictionary-size condition

$$r_n \leq r_{\max},\ r_{\max} \in \mathbb{N}^*, \tag{9}$$

i.e., the maximum dictionary size is $r_{\max} + M$, and (ii) the large-normalized-error condition

$$|\delta_n - \psi_n^{(n)}(z_n)|^2 > \epsilon |\psi_n^{(n)}(z_n)|^2,\ \epsilon \geq 0. \tag{10}$$

In Section III-B, we apply multikernel adaptive filtering to extract the dynamic structure of the agent dynamics.

III. Safety-aware Adaptive Reinforcement Learning

In this section, we present a safety-aware adaptive reinforcement learning framework.Throughout, we consider the following discrete-time deterministic nonlinear model of the nonstationary agent dynamics,

$$x_{n+1} - x_n = p_n(x_n, u_n) + f_n(x_n) + g_n(x_n) u_n, \tag{11}$$

where $p_n : \mathcal{X} \times \mathcal{U} \to \mathbb{R}^{n_x}$, $f_n : \mathcal{X} \to \mathbb{R}^{n_x}$, $g_n : \mathcal{X} \to \mathbb{R}^{n_x \times n_u}$. Hereafter, we regard $\mathcal{X} \times \mathcal{U}$ as the same as $\mathcal{Z} \subset \mathbb{R}^{n_x + n_u}$ under the one-to-one correspondence between $z := [x^\mathsf{T}, u^\mathsf{T}]^\mathsf{T} \in \mathcal{Z}$ and $(x, u) \in \mathcal{X} \times \mathcal{U}$, if there is no confusion.

A. Safe Maneuver: Discrete-time Control Barrier Function

Given a discrete-time exponential control barrier function B and $0 < \eta \leq 1$, the barrier certified safe control space at time instance n is define as

$$\mathcal{S}_n(x) := \{u_n \in \mathcal{U} | B(x_{n+1}) - B(x_n) \geq -\eta B(x_n)\}. \tag{12}$$

From Proposition 1, the set $\mathcal{C}$ defined in (2) is forward invariant and asymptotically stable if $u_n \in \mathcal{S}_n(x_n)$ for all $n \in \mathbb{N}^*$. As pointed out in [24], $\mathcal{S}_n(x_n) \subset \mathcal{U}$ is not a convex set in

general. To ensure global optimality of the solutions to the constrained feedback-controller optimization when updating a feedback controller (see Section III-D), we make the following moderate assumptions

Assumption 1. 1) *Control-affine agent dynamics: The agent dynamics is control affine, i.e.,* $p_n = 0$.

2) *Existence of Lipschitz continuous gradient of the barrier function: Given*

$$\mathcal{R} := \{(1-t)x_n + t(f_n(x_n) + g_n(x_n)u) | t \in [0,1], u \in \mathcal{U}\},$$

there exists a constant $\nu \geq 0$ *such that the gradient of the discrete-time exponential control barrier function* B*, denoted by* $\frac{\partial B(x)}{\partial x}$*, satisfies that*

$$\left\| \frac{\partial B(a)}{\partial x} - \frac{\partial B(b)}{\partial x} \right\|_{\mathbb{R}^{n_x}} \leq \nu \left\| a - b \right\|_{\mathbb{R}^{n_x}}, \ \forall a, b \in \mathcal{R}.$$

Then, the following theorem holds.

Theorem 1. *Under Assumptions 1.1 and 1.2, (3) is satisfied if* u_n *satisfies the following:*

$$\begin{aligned} &\frac{\partial B(x_n)}{\partial x}(f_n(x_n) + g_n(x_n)u_n) \\ &\geq -\eta B(x_n) + \frac{\nu}{2} \left\| f_n(x_n) + g_n(x_n)u_n \right\|^2_{\mathbb{R}^{n_x}}. \end{aligned} \tag{13}$$

Moreover, (13) defines a convex constraint for u_n.

Proof. See Appendix A. □

Theorem 1 essentially implies that, even when the gradient of B along the shift of x_n decreases steeply, (3) follows if (13) is satisfied. From Theorem 1, the set $\hat{\mathcal{S}}_n$, defined as

$$\begin{aligned} \hat{\mathcal{S}}_n := \{u_n \in \mathcal{U} | &\frac{\partial B(x_n)}{\partial x}(f_n(x_n) + g_n(x_n)u_n) \\ &\geq -\eta B(x_n) + \frac{\nu}{2} \left\| f_n(x_n) + g_n(x_n)u_n \right\|^2_{\mathbb{R}^{n_x}} \} \subset \mathcal{S}_n, \end{aligned} \tag{14}$$

is convex.

As witnessed in the literatures (e.g. [22]), an agent might encounter deadlock situations, where the constrained control keeps the agent remain in the same state, when control barrier certificates are employed. It is even possible that there is no safe control driving the agent from those states. However, an elaborative design of control barrier functions remedies this issue, as shown in the following example.

Example 1. *If the agent is nonholonomic, turning inward safe regions when approaching their boundary might be infeasible. To reduce the risk of such deadlock situations, control barrier functions may be designed as*

$$B(x) = \tilde{B}(x) - \upsilon\Gamma\left(\left|\theta - \operatorname{atan2}\left\{\frac{\partial \tilde{B}(x)}{\partial \mathrm{y}}, \frac{\partial \tilde{B}(x)}{\partial \mathrm{x}}\right\}\right|\right), \quad \upsilon > 0, \tag{15}$$

where the state $x = [\mathrm{x}, \mathrm{y}, \theta]$ *consists of the X position* x*, the Y position* y*, and the orientation* θ *of an agent from the world frame,* $\{x \in \mathcal{X} | \tilde{B}(x) \geq 0\}$ *is the original safe region, and* Γ *is a strictly increasing function. This control barrier function forces the agent turn inward the original safe region before reaching its boundaries.*

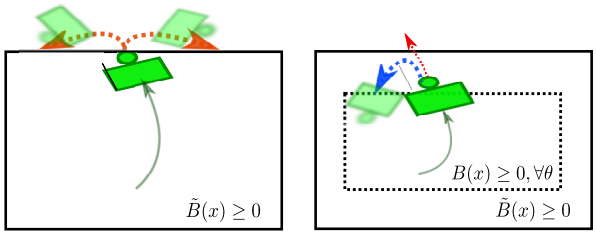

Fig. 3. An illustration of how a nonholonomic agent avoids deadlocks. When the orientation of the agent is not considered (i.e, $\tilde{B}(x)$ is the barrier function), there might be no safe control driving the agent from those states as the left figure shows. By taking into account the orientation (i.e, $B(x)$ is the barrier function), the agent turns inward the safe region before reaching its boundaries as the right figure shows.

An illustration of Example 1 is given in Figure 3.

B. Adaptive Model Learning–Capture Meaningful Structure

We have seen, in the previous subsection, that control-affine models are desirable to obtain barrier certified controllers efficiently. Here, we propose a model learning technique that also learns the dynamic structure. We assume that $n_x = 1$ for simplicity (we can employ n_x estimators if $n_x > 1$). Define $\psi_n(x_n, u_n) := p_n(x_n, u_n) + f_n(x_n) + g_n(x_n)u_n$, where $\psi_n : \mathcal{Z} \to \mathbb{R}$. We suppose that $p_n \in \mathcal{H}_p$, $f_n \in \mathcal{H}_f$, and $g_n \in \mathcal{H}_g$, where $\mathcal{H}_p$, $\mathcal{H}_f$, and $\mathcal{H}_g$ are RKHSs, thereby being able to employ kernel-based algorithms to learn a model. However, because the domains of p_n, f_n, and g_n are different, learning of the function ψ_n is infeasible at this stage. Let $\mathcal{H}_u$ be the RKHS associated with the reproducing kernel $\kappa(u, v) := u^\mathsf{T} v$, $u, v \in \mathcal{U}$, i.e a polynomial kernel with $c = 0$ and $d = 1$, and $\mathcal{H}_c := \{\varphi : \mathcal{U} \to \mathbb{R} | \exists \alpha \in \mathbb{R}, \varphi(u) = \alpha\} = \operatorname{span}\{\mathbf{1}\}$, where $\mathbf{1} : \mathcal{U} \to \{1\}$, the set of constant functions.

Proposition 2. *The space* $\mathcal{H}_c$ *is an RKHS associated with the reproducing kernel* $\kappa(u, v) = 1, \forall u, v \in \mathcal{U}$*, with the inner product defined as* $\langle \alpha\mathbf{1}, \beta\mathbf{1} \rangle_{\mathcal{H}_c} := \alpha\beta$, $\alpha, \beta \in \mathbb{R}$.

Proof. See Appendix B. □

Then, the following proposition implies that ψ_n can be approximated in the sum space of RKHSs defined later.

Proposition 3 ([49, Theorem 13])**.** *Let* $\mathcal{H}_1$ *and* $\mathcal{H}_2$ *be two RKHSs associated with the reproducing kernels* κ_1 *and* κ_2*. Then the completion of the tensor product of* $\mathcal{H}_1$ *and* $\mathcal{H}_2$*, denoted by* $\mathcal{H}_1 \otimes \mathcal{H}_2$*, is an RKHS associated with the reproducing kernel* $\kappa_1 \otimes \kappa_2$.

From Propositions 2 and 3, we can now assume that $\hat{f}_n \in \mathcal{H}_f \otimes \mathcal{H}_c$ and $\hat{g}_n \in \mathcal{H}_g \otimes \mathcal{H}_u$, where $\hat{f}_n : \mathcal{Z} \to \mathbb{R}$, $\hat{f}_n(x, u) := f_n(x)$, and $\hat{g}_n : \mathcal{Z} \to \mathbb{R}$, $\hat{g}_n(x, u) := g_n(x)u$, respectively. As such, ψ_n can be approximated in the RKHS $\mathcal{H}_\psi := \mathcal{H}_p + \mathcal{H}_f \otimes \mathcal{H}_c + \mathcal{H}_g \otimes \mathcal{H}_u$. By viewing the RKHS $\mathcal{H}_\psi$ as a sum space of $M \in \mathbb{N}^*$ RKHSs each of which is associated with the reproducing kernel κ_m, $m \in \overline{1, M}$, the current estimator $\psi_n^{(n)}$ can be evaluated at the current input $z_n := [x_n^\mathsf{T}, u_n^\mathsf{T}]^\mathsf{T} \in \mathcal{Z}$ as in (4).

Algorithm 1 Adaptive Model Learning

Requirement: $\lambda \in (0,2)$, $\rho \geq 0$ (precision parameter), $r_{\max} > 0$ (maximum-dictionary-size condition) $\epsilon \geq 0$ (large-normalized-error condition), $\mu \geq 0$ (sparsity parameter), $s_n \in \mathbb{N}^*$ (data size), $x_0 \in \mathcal{X}$, and $u_0 \in \mathcal{U}$
Initialization: $\mathcal{D}^-_{m,0} = \emptyset,\ h_{m,0} = [\,],\ m \in \overline{1,M}$
Output:
$p_n^{(n)}(z_n)$, $f_n^{(n)}(x_n)$, and $g_n^{(n)}(x_n)$ ▷ (16), (17), and (18)
for $n \in \mathbb{N}$ **do**
- Receive $x_n, x_{n+1} \in \mathcal{X}$ and $u_n \in \mathcal{U}$ ($\delta_n := x_{n+1} - x_n$)
- Check if the novelty conditions are satisfied ▷ (9), (10)
if Novelty conditions are satisfied **then**
Dictionary increment for $m \in \overline{1,M}$:
$\mathcal{D}_{m,n} = \mathcal{D}^-_{m,n} \cup \{\kappa_m(\cdot, [x_n^\mathsf{T}, u_n^\mathsf{T}]^\mathsf{T})\}$
Redefine $h_{m,n}$ as $[h_{m,n}^\mathsf{T}, 0]^\mathsf{T}$
end if
Update h_n ▷ (5), (7), and (8)
Eliminate the obsolete dictionary elements for $m \in \overline{1,M}$:
$\mathcal{D}^-_{m,n+1} := \{\kappa_m(\cdot, \tilde{z}_{m,j}) \in \mathcal{D}_{m,n} | h_{m,n+1}^\mathsf{T} e_j \neq 0\}$
Discard the zero entries of $h_{m,n+1}$
end for

Suppose that the RKHSs $\mathcal{H}_p$, $\mathcal{H}_f$, and $\mathcal{H}_g$ can be expressed as the sum spaces of $M_p, M_f, M_g \in \mathbb{N}^*$ RKHSs,where $M_p + M_f + M_g = M$. The evaluations of the current estimators $p_n^{(n)}$ and $f_n^{(n)}$ at the current inputs $z_n := [x_n^\mathsf{T}, u_n^\mathsf{T}]^\mathsf{T}$ and x_n are then given by

$$p_n^{(n)}(z_n) = \sum_{m=1}^{M_p} h_{m,n}^\mathsf{T} k_{m,n}(z_n), \tag{16}$$

$$f_n^{(n)}(x_n) = \hat{f}_n^{(n)}(z_n) = \sum_{m=M_p+1}^{M_p+M_f} h_{m,n}^\mathsf{T} k_{m,n}(z_n). \tag{17}$$

In order to use the learned model in combination with control barrier functions, each entry of the vector $g_n^{(n)}(x_n)$ (the current estimate of $g_n(x_n)$) is required. Assume, without loss of generality, that $\{e_i\}_{i \in \overline{1,n_u}} \subset \mathcal{U}$ (this is always possible for $\mathcal{U} \neq \emptyset$ by transforming the coordinate of control inputs and reduce the dimension n_u if necessary). Then, the ith entry of the vector $g_n^{(n)}(x_n)$ is given by

$$\begin{aligned} g_n^{(n)}(x_n)e_i &= \hat{g}_n^{(n)}(x_n, e_i) \\ &= \sum_{m=M_p+M_f+1}^{M} h_{m,n} k_{m,n}([x_n^\mathsf{T}, e_i^\mathsf{T}]^\mathsf{T}). \end{aligned} \tag{18}$$

The proposed model learning algorithm is summarized in Algorithm 1. We conclude this subsection by giving the following remarks.

Remark 1. *The following theorem ensures that ψ_n can be uniquely decomposed into p_n, $\hat{f}_n$, and $\hat{g}_n$.*

Theorem 2. *Assume that $\mathcal{X}$ and $\mathcal{U}$ have nonempty interiors. Assume also that $\mathcal{H}_p$ is a Gaussian RKHS. Then, $\mathcal{H}_\psi$ is the direct sum of $\mathcal{H}_p$, $\mathcal{H}_f \otimes \mathcal{H}_c$, and $\mathcal{H}_g \otimes \mathcal{H}_u$, i.e, the intersection of any two of the RKHSs $\mathcal{H}_p$, $\mathcal{H}_f \otimes \mathcal{H}_c$, and $\mathcal{H}_g \otimes \mathcal{H}_u$ is $\{0\}$.*

Proof. See Appendix C. □

In most of the kernel-based methods such as Gaussian processes and the kernel recursive least mean squares algorithm [50], tractability of the inner product between kernel functions is an essential property (e.g. for computing the Gram matrix). In a sum space of RKHSs, the inner product has no closed form in general. Nevertheless, from Theorem 2, the inner product is computed as

$$\begin{aligned} &\langle \psi_1, \psi_2 \rangle_{\mathcal{H}_\psi} \\ &= \langle p_1, p_2 \rangle_{\mathcal{H}_p} + \left\langle \hat{f}_1, \hat{f}_2 \right\rangle_{\mathcal{H}_f \otimes \mathcal{H}_c} + \langle \hat{g}_1, \hat{g}_2 \rangle_{\mathcal{H}_g \otimes \mathcal{H}_u}, \\ &\psi_i := p_i + \hat{f}_i + \hat{g}_i \in \mathcal{H}_\psi,\ i \in \overline{1,2}. \end{aligned}$$

Remark 2. *By using a sparse optimization for the coefficient vector $h_n \in \mathbb{R}^{r_n}$, we wish to extract a structure of the model. In the present study, the term $p_n^{(n)}$ is desired to be dropped off, when the true agent dynamics is control affine. To effectively achieve a compact representation of the model, it might be required to appropriately weigh the kernel functions to include some preferences on a structure of the model. The following proposition implies that the resulting kernels are still reproducing kernels.*

Proposition 4 ([29, Theorem 2])**.** *Let $\kappa : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ be the reproducing kernel of an RKHS $(\mathcal{H}, \langle \cdot, \cdot \rangle_{\mathcal{H}})$. Then, $\tau\kappa(z,w),\ z,w \in \mathcal{Z}$ for arbitrary $\tau > 0$ is the reproducing kernel of the RKHS $(\mathcal{H}_\tau, \langle \cdot, \cdot \rangle_{\mathcal{H}_\tau})$ with the inner product $\langle z, w \rangle_{\mathcal{H}_\tau} := \tau^{-1} \langle z, w \rangle_{\mathcal{H}},\ z, w \in \mathcal{Z}$.*

C. Adaptive Reinforcement Learning

We reformulate the (action-)value function approximation problem to enable to directly apply any kernel-based method, including multikernel adaptive filtering. The Bellman equation of a feedback controller $\phi : \mathcal{X} \to \mathcal{U}$ for the value function is given in (1). Assume the target feedback controller ϕ is deterministic, and let $\mathcal{H}_V$ be an RKHS containing the value function V^ϕ. By slightly modifying (1), we obtain

$$V_n^\phi(x_n) - \gamma V_n^\phi(x_{n+1}) = R_n(x_n, \phi(x_n)), \tag{19}$$

where the value function and the instantaneous-reward function are now time-dependent. Define a function $\psi_n : \mathcal{X}^2 \to \mathbb{R}$, where $\mathbb{R}^{2n_x} \supset \mathcal{X}^2 = \mathcal{X} \times \mathcal{X}$, as

$$\psi_n([x^\mathsf{T}, y^\mathsf{T}]^\mathsf{T}) := V_n^\phi(x) - \gamma V_n^\phi(y),\ \forall x, y \in \mathcal{X}. \tag{20}$$

Then, (19) is reformulated as

$$\psi_n([x_n^\mathsf{T}, x_{n+1}^\mathsf{T}]^\mathsf{T}) = R_n(x_n, \phi(x_n)).$$

As such, solving the Bellman equation comes down to the iterative nonlinear function estimation with the input-output pairs $\{([x_n^\mathsf{T}, x_{n+1}^\mathsf{T}]^\mathsf{T}, R_n(x_n, \phi(x_n)))\}_{n \in \mathbb{N}}$. In the case of the action-value function, the Bellman equation of a feedback controller $\phi : \mathcal{X} \to \mathcal{U}$ is given by

$$Q_n^\phi(x_n, u_n) = \gamma Q_n^\phi(x_{n+1}, \phi(x_{n+1})) + R_n(x_n, u_n), \tag{21}$$

where $Q_n^\phi \in \mathcal{H}_Q$ denotes the true action-value function with respect to ϕ, contained in an RKHS $\mathcal{H}_Q$, at time instance n.

By defining a function $\psi_n : \mathcal{Z}^2 \to \mathbb{R}$, where $\mathbb{R}^{2(n_x+n_u)} \supset \mathcal{Z}^2 = \mathcal{Z} \times \mathcal{Z}$, as

$$\psi_n([x^\mathsf{T}, u^\mathsf{T}, y^\mathsf{T}, v^\mathsf{T}]^\mathsf{T}) := Q_n^\phi(x,u) - \gamma Q_n^\phi(y,v), \\ x, y \in \mathcal{X},\ u, v \in \mathcal{U},$$

the Bellman equation in (21) is solved via iterative nonlinear function estimation with the input-output pairs $\{([x_n^\mathsf{T}, u_n^\mathsf{T}, x_{n+1}^\mathsf{T}, \phi(x_{n+1})^\mathsf{T}]^\mathsf{T}, R_n(x_n,u_n))\}_{n\in\mathbb{N}}$.

Theorem 3. *Suppose that $\mathcal{H}_Q$ is an RKHS associated with the reproducing kernel $\kappa^Q(\cdot,\cdot) : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$. Then,*

$$\mathcal{H}_\psi := \{\varphi | \varphi([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) = \varphi^Q(z) - \gamma\varphi^Q(w), \gamma \in (0,1) \\ \varphi^Q \in \mathcal{H}_Q,\ z, w \in \mathcal{Z}\},$$

is also an RKHS with the inner product defined by

$$\langle \varphi_1, \varphi_2 \rangle_{\mathcal{H}_\psi} := \left\langle \varphi_1^Q, \varphi_2^Q \right\rangle_{\mathcal{H}_Q}, \tag{22}$$
$$\varphi_i([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) := \varphi_i^Q(z) - \gamma\varphi_i^Q(w),\ \forall z, w \in \mathcal{Z},\ i \in \overline{1,2}.$$

The reproducing kernel of the RKHS $\mathcal{H}_\psi$ is given by

$$\kappa([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}, [\tilde{z}^\mathsf{T}, \tilde{w}^\mathsf{T}]^\mathsf{T}) \\ := \left(\kappa^Q(z,\tilde{z}) - \gamma\kappa^Q(z,\tilde{w})\right) \\ - \gamma\left(\kappa^Q(w,\tilde{z}) - \gamma\kappa^Q(w,\tilde{w})\right),\ z, w, \tilde{z}, \tilde{w} \in \mathcal{Z}.$$

Proof. See Appendix D. □

From Theorem 3, any kernel-based method can be applied by assuming that $\psi_n \in \mathcal{H}_\psi$. Suppose that multikernel adaptive filtering is employed. Given a sum space of M RKHSs each of which is associated with the reproducing kernel κ_m^Q, $m \in \overline{1,M}$, we define the reproducing kernels $\kappa_m([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}, [\tilde{z}^\mathsf{T}, \tilde{w}^\mathsf{T}]^\mathsf{T}) := \left(\kappa_m^Q(z,\tilde{z}) - \gamma\kappa_m^Q(z,\tilde{w})\right) - \gamma\left(\kappa_m^Q(w,\tilde{z}) - \gamma\kappa_m^Q(w,\tilde{w})\right)$, $m \in \overline{1,M}$, and their corresponding RKHSs as in Theorem 3. Provided the dictionaries $\mathcal{D}_{m,n} := \{\kappa_m(\cdot, [\tilde{z}_{m,j}^\mathsf{T}, \tilde{w}_{m,j}^\mathsf{T}]^\mathsf{T})\}_{j\in\overline{1,r_{m,n}}}$, $m \in \overline{1,M}$, the estimator $\psi_n^{(n)}$ is then expressed as

$$\psi_n^{(n)}([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) := h_n^\mathsf{T} k_n([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) \\ = \sum_{m=1}^M h_{m,n}^\mathsf{T} k_{m,n}([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}),\ z, w \in \mathcal{Z}, \tag{23}$$

where $k_{m,n}([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T})^\mathsf{T} e_j := \kappa_m\left([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}, [\tilde{z}_{m,j}^\mathsf{T}, \tilde{w}_{m,j}^\mathsf{T}]^\mathsf{T}\right)$, $j \in \overline{1, r_{m,n}}$, from which we obtain the estimator of the action-value function $Q_n^{\phi^{(n)}} \in \mathcal{H}_Q$ as

$$Q_n^{\phi^{(n)}}(z) := h_n^\mathsf{T} k_n^Q(z) = \sum_{m=1}^M h_{m,n}^\mathsf{T} k_{m,n}^Q(z), \tag{24}$$

where $k_n^Q(z) := \left[k_{1,n}^Q(z)^\mathsf{T} k_{2,n}^Q(z)^\mathsf{T} \cdots k_{M,n}^Q(z)^\mathsf{T}\right]^\mathsf{T} \in \mathbb{R}^{r_n}$, and $k_{m,n}^Q(z)^\mathsf{T} e_j := \kappa_m^Q\left(z, \tilde{z}_{m,j}^\mathsf{T}\right) - \gamma\kappa_m^Q\left(z, \tilde{w}_{m,j}\right)$, $j \in \overline{1, r_{m,n}}$.

Resulting action-value function approximation algorithm is summarized in Algorithm 2.

Remark 3. *Employing the action-value function enables to use random control inputs instead of the target feedback controller ϕ for exploration.*

Algorithm 2 Adaptive Action-value Function Approximation Algorithm

Requirement: Assumption 1, κ_m^Q defined in (26), $\lambda \in (0,2)$, $\rho \geq 0$ (precision parameter), $r_{\max} > 0$ (maximum-dictionary-size condition) $\epsilon \geq 0$ (large-normalized-error condition), $\mu \geq 0$ (sparsity parameter), $s_n \in \mathbb{N}^*$ (data size), $x_0 \in \mathcal{X}$ and $u_0 \in \mathcal{U}$
Initialization: $\mathcal{D}_{m,0}^- = \emptyset$, $h_{m,0} = [\ \ ]$, $m \in \overline{1,M}$
Output: $Q_n^{\phi^{(n)}}(z_n)$ ▷ (24)
for $n \in \mathbb{N}$ **do**
  - Receive $x_n, x_{n+1} \in \mathcal{X}$, and $\delta_n := R_n(x_n,u_n) \in \mathbb{R}$
  - Obtain $\phi(x_{n+1}) \in \hat{\mathcal{S}}_n(x_{n+1})$ ▷ Algorithm 3
  if Random Exploration **then**
    Generate random control input $u_{n+1} \in \hat{\mathcal{S}}_n(x_{n+1})$ ▷ Algorithm 1, Theorem 1
  else
    $u_{n+1} = \phi(x_{n+1})$ ▷ Algorithm 3
  end if
  - Check if the novelty conditions are satisfied ▷ (9), (10)
  if Novelty conditions are satisfied **then**
    Dictionary increment for $m \in \overline{1,M}$:
    $\mathcal{D}_{m,n} = \mathcal{D}_{m,n}^- \cup \{\kappa_m(\cdot, [x_n^\mathsf{T}, u_n^\mathsf{T}, x_{n+1}^\mathsf{T}, \phi(x_{n+1})^\mathsf{T}]^\mathsf{T})\}$
    Redefine $h_{m,n}$ as $[h_{m,n}^\mathsf{T}, 0]^\mathsf{T}$
  end if
  Update h_n ▷ (5), (7), and (8)
  Eliminate the obsolete dictionary elements for $m \in \overline{1,M}$:
  $\mathcal{D}_{m,n+1}^-$
  $:= \{\kappa_m(\cdot, [\tilde{z}_{m,j}^\mathsf{T}, \tilde{w}_{m,j}^\mathsf{T}]^\mathsf{T}) \in \mathcal{D}_{m,n} | h_{m,n+1}^\mathsf{T} e_j \neq 0\}$
  Discard the zero entries of $h_{m,n+1}$
end for

Remark 4. *When the coefficient vector h_n for the estimator $\psi_n^{(n)}$ in (23) is monotonically approaching to a optimal point h^* in the Euclidean norm sense, so is the coefficient vector for the (action-)value function because the same coefficient vector is used to estimate ψ_n and Q_n^ϕ (see (23) and (24)). Suppose we employ a method in which $\psi_n^{(n)}$ is monotonically approaching to a optimal function ψ_n^* in the Hilbertian norm sense. Then, the following corollary implies that an estimator of the action-value function also satisfies the monotonicity.*

Corollary 1. *Let $\mathcal{H}_\psi \ni \psi_n^{(n)}(z,w) := Q_n^{\phi^{(n)}}(z) - \gamma Q_n^{\phi^{(n)}}(w)$ and $\mathcal{H}_\psi \ni \psi_n^*(z,w) := Q_n^{\phi^*}(z) - \gamma Q_n^{\phi^*}(w)$, $z, w \in \mathcal{Z}$, where $Q_n^{\phi^{(n)}}, Q_n^{\phi^*} \in \mathcal{H}_Q$. Then, if $\psi_n^{(n)}$ is approaching to ψ_n^* in the Hilbertian norm sense, i.e.,*

$$\left\|\psi_n^{(n+1)} - \psi_n^*\right\|_{\mathcal{H}_\psi} \leq \left\|\psi_n^{(n)} - \psi_n^*\right\|_{\mathcal{H}_\psi},$$

it holds that

$$\left\|Q_n^{\phi^{(n+1)}} - Q_n^{\phi^*}\right\|_{\mathcal{H}_Q} \leq \left\|Q_n^{\phi^{(n)}} - Q_n^{\phi^*}\right\|_{\mathcal{H}_Q}.$$

Proof. See Appendix E. □

D. Safety-aware Feedback-controller Update

Given current feedback controller $\phi : \mathcal{X} \to \mathcal{U}$, assume that the action-value function Q_n^ϕ with respect to ϕ at time instance

Algorithm 3 Feedback-controller Update

Requirement: Assumption 1, $N_f \in \mathbb{N}^*$ (update frequency)
Initialization: $\phi = 0$
for $n \in \mathbb{N}$ **do**
  if $n \bmod N_f = 0$ **then**
    Update ϕ ▷ (27), Algorithm 1, and Theorem 1
    $\phi = \phi^+$
  end if
end for

n is available. Then, the feedback controller ϕ^+ given by

$$\phi^+(x) := \underset{u \in \hat{\mathcal{S}}_n(x)}{\operatorname{argmax}} \left[Q_n^{\phi}(x, u) \right], \tag{25}$$

where $\hat{\mathcal{S}}_n(x) \subset \mathcal{U}$ is defined in (14), is well-known (e.g. [51]) to satisfy that $Q_n^{\phi}(x, \phi(x)) \leq Q_n^{\phi^+}(x, \phi^+(x))$, where $Q_n^{\phi^+}$ is the action-value function with respect to ϕ^+ at time instance n. In practice, we use the estimator of Q_n^{ϕ} because the exact function Q_n^{ϕ} is unavailable. For example, the action-value function is estimated over $N_f \in \mathbb{N}^*$ iterations, and the feedback controller is updated every N_f iterations. To obtain analytical solutions for (25), we follow the arguments in [35]. Suppose that $Q_n^{\phi^{(n)}}$ is given by (24). We define the reproducing kernel κ_m^Q, $m \in \overline{1, M}$ as the tensor kernel given by

$$\kappa_m^Q([x^\mathsf{T}, u^\mathsf{T}]^\mathsf{T}, [y^\mathsf{T}, v^\mathsf{T}]^\mathsf{T}) := \kappa_m^x(x, y)\kappa_m^u(u, v), \tag{26}$$

where $\kappa_m^u(u, v)$ is, for example, defined by

$$\kappa_m^u(u, v) := 1 + \frac{1}{4}(u^\mathsf{T} v).$$

Then, (25) becomes

$$\phi^+(x) := \underset{u \in \hat{\mathcal{S}}_n(x)}{\operatorname{argmax}} \left[h_n^\mathsf{T} k_n^Q([x^\mathsf{T}, u^\mathsf{T}]^\mathsf{T}) \right], \tag{27}$$

where the target value being maximized is linear to u at x. Therefore, convexity of the set $\hat{\mathcal{S}}_n(x) \subset \mathcal{U}$ implies that an optimal solution to (27) is guaranteed to be globally optimal, ensuring the monotonic improvement of the feedback controller.

The proposed feedback-controller update is summarized in Algorithm 3, where mod stands for the modulo operation.

IV. Experimental Results

The proposed learning framework is implemented on a *brushbot*, which has highly nonlinear, nonholonomic, and complex dynamics (see Figure 4). The objective of this experiment is to find a feedback controller driving the brushbot to the origin, while restricting the region of exploration. The experiment is conducted at the Robotarium, a remotely accessible robot testbed at Georgia institute of technology [52].

A. Experimental Condition

The experimental conditions for model learning, reinforcement learning, control barrier functions, and their parameter settings are presented below.

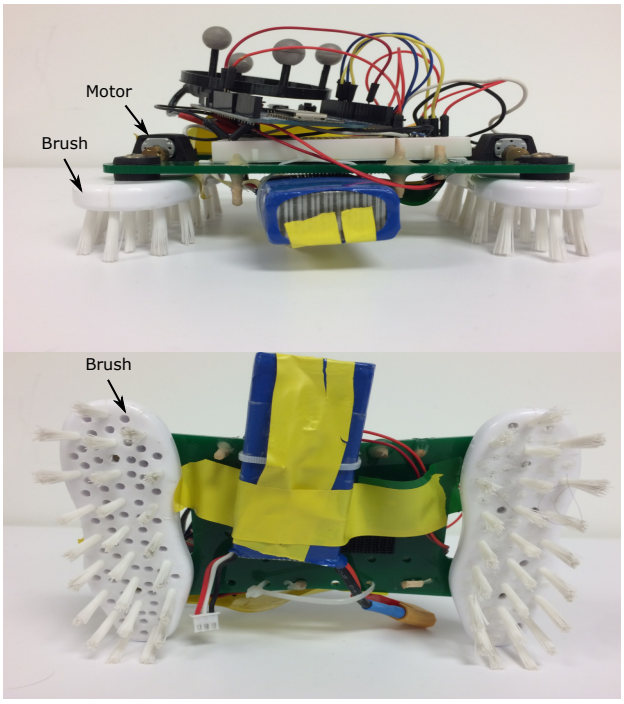

Fig. 4. A picture of the brushbot used in the experiment. Vibrations of the two motors propagate to the two brushes, driving the brushbot. Control inputs are of two dimension each of which corresponds to the rotational speed of a motor.

1) Model learning: The state $x = [\mathrm{x}, \mathrm{y}, \theta]^\mathsf{T}$ consists of the X position x, Y position y, and the orientation $\theta \in [-\pi, \pi]$ of the brushbot from the world frame. The exact positions and the orientation are recorded by a motion capture system every 0.3 second. A control input u is of two dimension each of which corresponds to the rotational speed of a motor. To improve the learning efficiency and reduce the total learning time required, we identify the most significant dimension and reduce the dimensions to learn. The sole input variable of p_n, f_n, and g_n, for the shifts of x and y, is assumed to be θ, and the shift of θ is assumed to be constant over the state, and hence depends on nothing but control inputs (see Section IV-A.4). The brushbot used in the present study is nonholonomic, i.e., it can only go forward, and positive control inputs basically drive the brushbot in the same way as negative control inputs. As such, we use the rotational speeds of the motors as the control inputs Moreover, to eliminate the effect of static frictions on the model, we assume that the zero control input given to the algorithm actually generate some minimum control inputs u_δ to the motors, i.e., the actual maximum control inputs to the motors are given by $u_{\max} + u_\delta$, where $u_{\max}$ is the maximum control input fed to the algorithm.

2) Reinforcement learning: The state for the action-value function approximation consists of the distance $\left\| [\mathrm{x}, \mathrm{y}]^\mathsf{T} \right\|_{\mathbb{R}^2}$ from the origin and the orientation θ. The instantaneous reward is given by

$$R_n(x, u) = -\left\| [\mathrm{x}, \mathrm{y}]^\mathsf{T} \right\|_{\mathbb{R}^2}^2 + 2, \ \forall n \in \mathbb{N} \tag{28}$$

where the constant is added to prevent the resulting value of explored states from becoming negative, i.e lower than the

value outside of the region of exploration.

3) Discrete-time control barrier certificate: Control barrier certificates are used to limit the region of exploration to a recutangular area: $\mathrm{x} \in [-\mathrm{x}_{\max}, \mathrm{x}_{\max}]$, $\mathrm{y} \in [-\mathrm{y}_{\max}, \mathrm{y}_{\max}]$, where $\mathrm{x}_{\max} > 0$ and $\mathrm{y}_{\max} > 0$. Because the brushbot can only go forward, we employ the following four barrier functions:

$$B_1(x) = \mathrm{x}_{\max} - \mathrm{x} - \upsilon \left|\theta + \pi\right|,$$
$$B_2(x) = \mathrm{x} + \mathrm{x}_{\max} - \upsilon \left|\theta\right|,$$
$$B_3(x) = \mathrm{y}_{\max} - \mathrm{y} - \upsilon \left|\theta + \frac{\pi}{2}\right|,$$
$$B_4(x) = \mathrm{y} + \mathrm{y}_{\max} - \upsilon \left|\theta - \frac{\pi}{2}\right|,$$

(see Example 1). Note that those functions satisfy Assumption 1.2 and the Lipschitz constant ν is zero except at around $\theta = -\frac{\pi}{2}, 0, \frac{\pi}{2}, \pi$.

4) Parameter settings: The parameter setting is summarized in Table I. Five Gaussian kernels with different scale parameters σ are employed in the action-value function approximation (i.e, $M = 5$), and six Gaussian kernels are employed in model learning for x and y (i.e, $M_p = M_f = M_g = 6$). In model learning for θ, we define $\mathcal{H}_p := \{\varphi : \mathcal{Z} \to \mathbb{R} | \exists \alpha \in \mathbb{R}, \varphi(z) = \alpha\}$ and $\mathcal{H}_f = \mathcal{H}_g := \{\varphi : \mathcal{X} \to \mathbb{R} | \exists \alpha \in \mathbb{R}, \varphi(x) = \alpha\}$ (i.e, $M_p = M_f = M_g = 1$).

The kernels of $\mathcal{H}_p$ and $\mathcal{H}_f$ are weighed by $\tau = 0.1$ in model learning (see Remark 2).

5) Procedure: The time interval (duration of one iteration) for learning is 0.3 seconds, and random explorations are conducted for the first 300 seconds corresponding to 1000 iterations. While exploring, the model learning algorithm adaptively learns a model whose control-affine terms, i.e. $f_n(x) + g_n(x)u$, is used in combination with barrier certificates. Although barrier functions employed in the experiment reduce deadlock situations, the brushbot is forced to turn inward the region of exploration when a deadlock is detected. Note that the barrier certificates are intentionally violated in such a case. The feedback controller is updated every 50 seconds. After 300 seconds, we stop learning a model and the action-value function, and the feedback controller replaces random explorations. The brushbot is forced to stop when it enters into the circle of radius 0.2 centered at the origin. When the brushbot is driven close to the origin and enters this circle, it is pushed away from the origin to see if it returns to the origin again (see Figure 10).

TABLE I

SUMMARY OF THE PARAMETER SETTINGS

General Setting		
(Parameter)	(Description)	(Value)
$\mathrm{x}_{\max}$	maximum X position	1.2
$\mathrm{y}_{\max}$	maximum Y position	1.2
η	Barrier-function parameter	0.1
υ	coefficient in barrier functions	0.1
u_δ	actual minimum control	0.4
$u_{\max}$	maximum control input	0.623
Model Learning for x **and** y		
(Parameter)	(Description)	(Value)
λ	step size	0.3
s_n	data size	5
μ	regularization parameter	0.0001
ρ	precision parameter	0.001
ϵ	large-normalized-error	0.1
$r_{\max}$	maximum-dictionary-size	500
σ	scale parameters	$\{10, 5, 2, 1, 0.5, 0.2\}$
Model Learning for θ		
(Parameter)	(Description)	(Value)
λ	step size	0.03
s_n	data size	10
μ	regularization parameter	0
ρ	precision parameter	0.01
ϵ	large-normalized-error	0.1
$r_{\max}$	maximum-dictionary-size	3
Action-value Function Approximation		
(Parameter)	(Description)	(Value)
λ	step size	0.3
s_n	data size	10
μ	regularization parameter	0.0001
ρ	precision parameter	0.05
ϵ	large-normalized-error	0.1
$r_{\max}$	maximum-dictionary-size	2000
γ	discount factor	0.95
σ	scale parameters	$\{10, 5, 2, 1, 0.5\}$

B. Results

Figure 5 plots $p_n^{(n)}([x^\mathsf{T}, [0,0]]^\mathsf{T})$, $f_n^{(n)}(x)$, and $g_n^{(n)}(x)e_i$, $i \in \overline{1,2}$, for x and y at $n = 1000$. Recall that these functions only depend on θ in this experiment to improve the learning efficiency. For the shift of θ, the estimators are constant over the state (see Section IV-A.1), and the result is $g_n^{(n)}(x)e_1 = 1.38$, $g_n^{(n)}(x)e_2 = -0.77$, and $p_n^{(n)}([x^\mathsf{T}, [0,0]]^\mathsf{T}) = f_n^{(n)}(x) = 0$ at $n = 1000$. As can be seen in Figure 5, $p_n^{(n)}([x^\mathsf{T}, [0,0]^\mathsf{T}]^\mathsf{T})$ is almost zero and so is $f_n^{(n)}(x)$, implying that the proposed algorithm successfully drops off irrelevant structural components of a model.

Figure 6 plots the trajectory of the brushbot while exploring (i.e. X,Y positions from $n = 0$ to $n = 1000$). It is observed that the brushbot remains in the region of exploration ($\mathrm{x} \in [-1.2, 1.2]$ and $\mathrm{y} \in [-1.2, 1.2]$) most of the time. Moreover, the values of barrier functions B_i, $i \in \overline{1,4}$, for the whole trajectory are plotted in Figure 7. Even though some violations of safety are seen in the figure, the brushbot returns to the safe region before large violations occur. Despite unknown, highly complex and possibly nonstationary system, the proposed safety-aware learning framework is shown to work efficiently.

Figure 8 plots the trajectories of the optimal feedback controller learned by the brushbot. Once the optimal feedback controller replaces random explorations, the brushbot returns to the origin until $n = 1016$ as the first figure shows. The brushbot is pushed by a sweeper at time instance $n = 1031, 1075, 1101, 1128, 1181$, and $n = 1230$, and the trajectories of the brushbot after being pushed at $n = 1031, 1075, 1101$ are also shown in Figure 8. Dashed lines in the last figure indicates the time when the brushbot is pushed away. Given relatively short learning time and that no simulator is used, the brushbot learns the desirable behavior sufficiently well.

Figure 9 plots the shape of $Q_n^{\phi^{(n)}}\left(\left[\left\|[\mathrm{x}, \mathrm{y}]^\mathsf{T}\right\|_{\mathbb{R}^2}, \theta\right]^\mathsf{T}, [0,0]^\mathsf{T}\right)$ over X,Y positions at $n = 1000$. It is observed that when the control input is zero (i.e., when the brushbot basically does not move), the vicinity

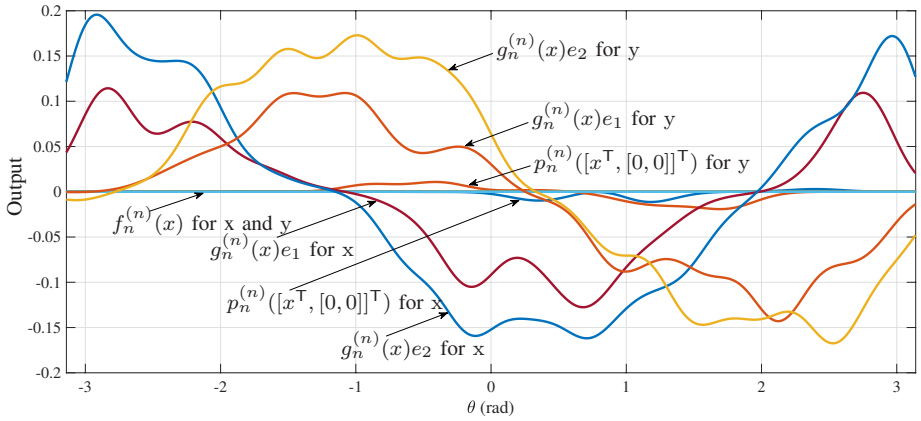

Fig. 5. Estimated output of the model estimator at $u = [0, 0]^\mathsf{T}$ and $n = 1000$ over the orientation θ. Irrelevant structures such as $p_n^{(n)}$ and $f_n^{(n)}$ dropped off successfully.

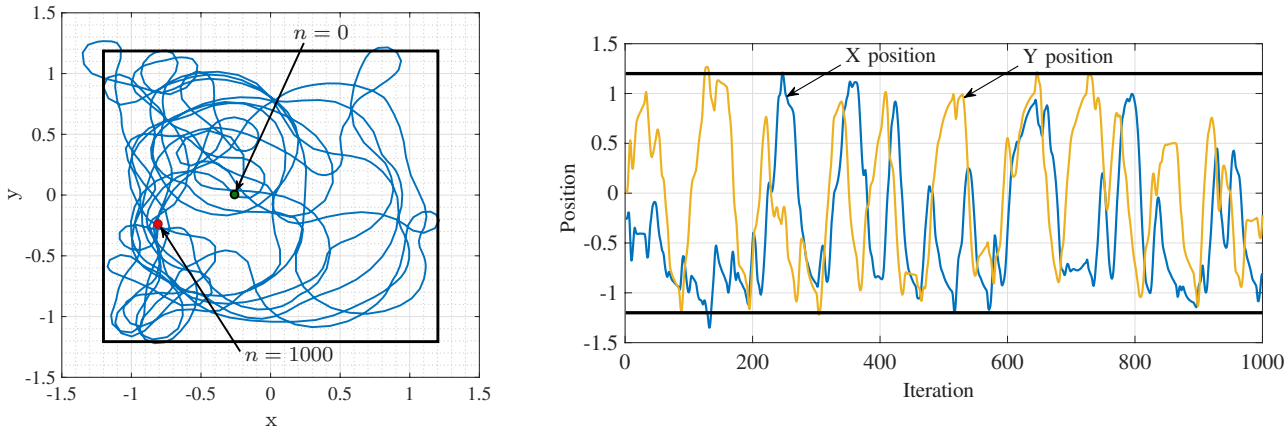

Fig. 6. The left figure shows the trajectory of the brushbot while exploring, and the right figure shows X,Y positions over iterations. The region of exploration is limited to x $\in [-1.2, 1.2]$ and y $\in [-1.2, 1.2]$. The brushbot remains in the region most of the time.

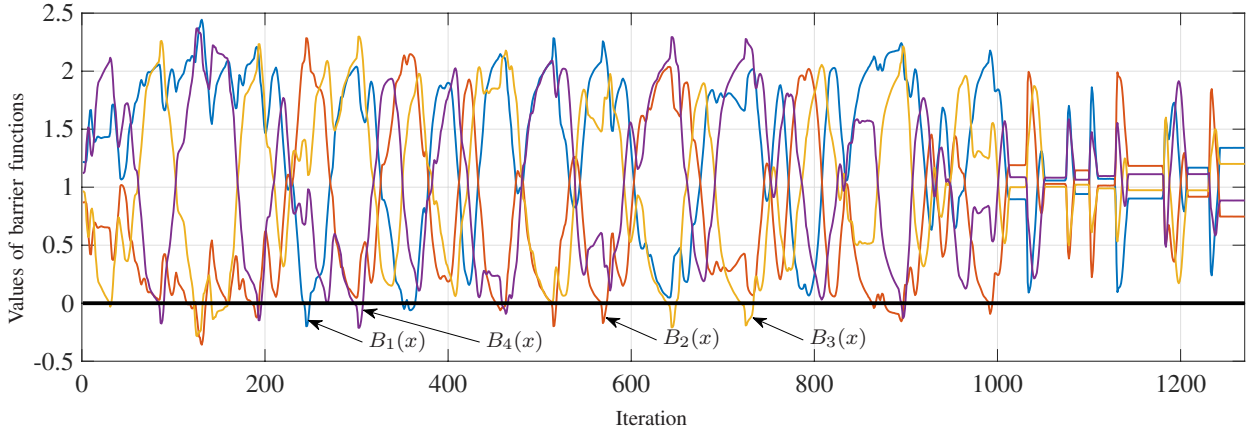

Fig. 7. The values of four control barrier functions employed in the experiment for the whole trajectory. Even though some violations of safety are seen, the brushbot returns to the safe region before large violations occur. The nonholonomic brushbot adaptively learns a model and how to turn inward the region of exploration before reaching the boundaries of the region of exploration.

of the origin has the highest value, which is reasonable.

Finally, Figure 10 shows two trajectories of the brushbot returning to the origin by using the action-value function saved at $n = 1000$. After being pushed away from the origin, the brushbot successfully returns to the origin again.

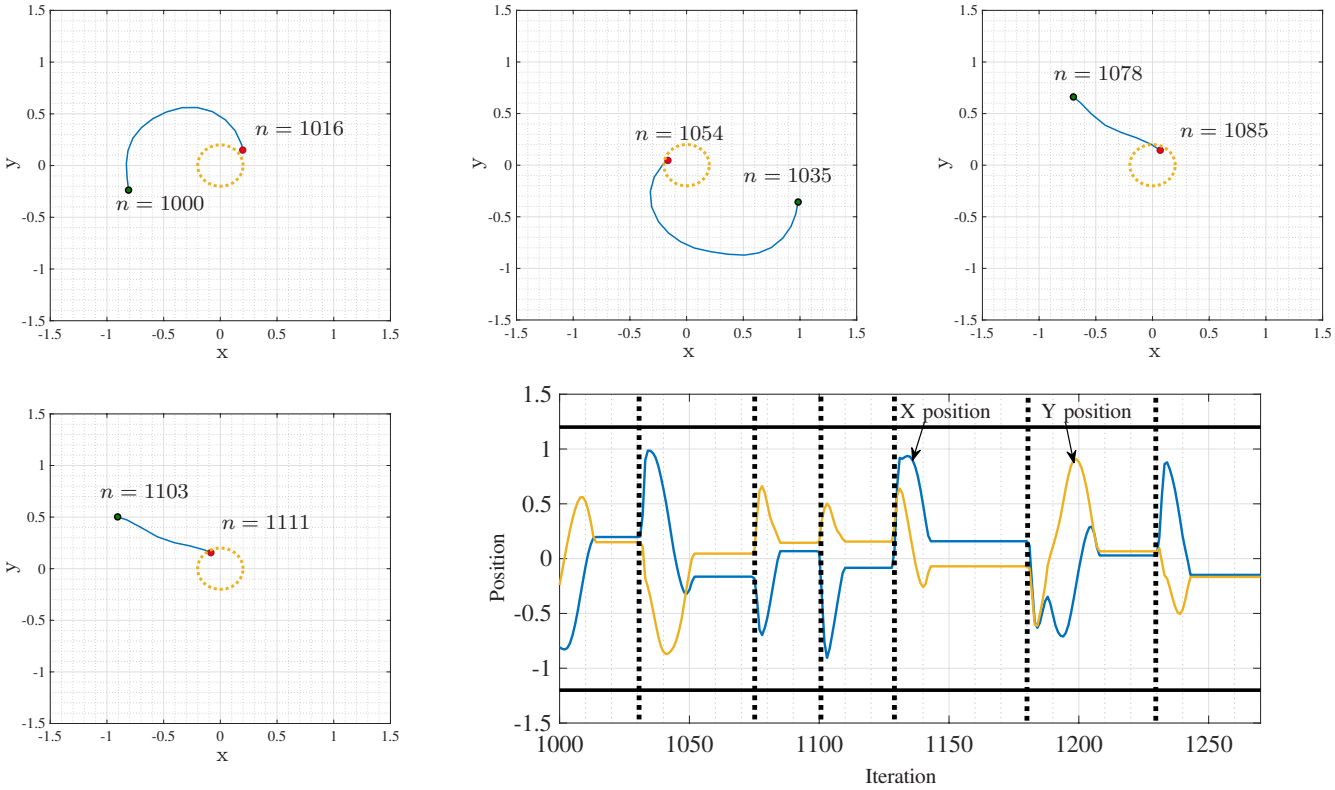

Fig. 8. Trajectories of the optimal feedback controller learned by the brushbot. The optimal feedback controller replaces random explorations at $n = 1000$, and the brushbot returns to the origin until $n = 1016$ (first figure). The brushbot is pushed by a sweeper at time instance $n = 1031, 1075, 1101, 1128, 1181$, and $n = 1230$. Dashed lines in the last figure indicates the time when the brushbot is pushed away. The brushbot learns the desirable behavior sufficiently well.

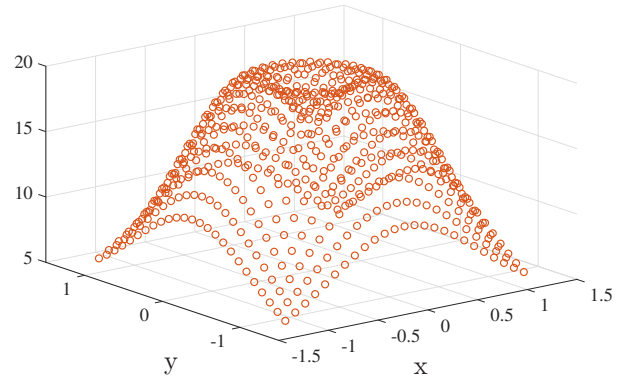

Fig. 9. The shape of the action-value function over X,Y positions at the control input $u = 0$ and $n = 1000$. The vicinity of the origin has the highest value when the control input is zero.

C. Discussion

One of the challenges of the experiment is that no initial data or simulators are available. Despite the fact that the brushbot with highly complex system has to learn an optimal controller while dealing with safety by employing adaptive model learning, the proposed learning framework works well in the real world. If some initial data or knowledges about a model are available, it is also possible to combine the proposed framework with other methods designed for stationary dynamics. For model learning and the action-value function approximation, multikernel adaptive filtering might be replaced by other kernel-based methods as well.

V. Conclusion

The learning framework presented in this paper successfully tied model learning, reinforcement learning, and barrier certificates, enabling safety-aware reinforcement learning for unknown, highly nonlinear, nonholonomic, and possibly nonstationary agent dynamics. The proposed model learning algorithm is able to capture a structure of the agent dynamics by employing a sparse optimization. The resulting model has preferable structure for preserving efficient computations of barrier certificates. A set of mild conditions ensuring convexity of the barrier certified safe control space is presented. Convexity of the safe control space and an appropriate design of kernel functions for the action-value function guarantee the global optimality of solutions to the feedback-controller update, which ensures the monotonic improvement of a feedback controller. In addition, the (action-)value function approximation problem is appropriately reformulated so that kernel-based methods including multikernel adaptive filtering can be directly applied. The experimental result shows the efficacy of the proposed learning framework in the real world.

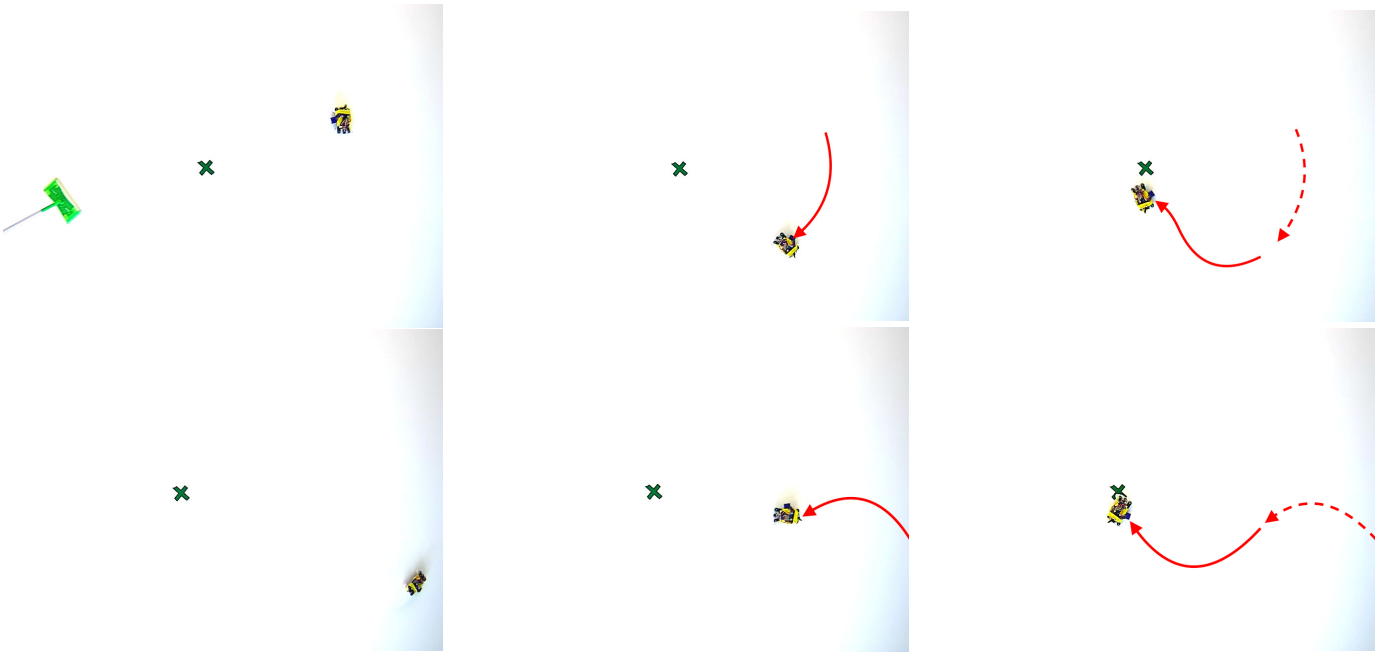

Fig. 10. Two trajectories of the brushbot returning to the origin by using the action-value function saved at $n = 1000$. Red arrows show the trajectories. After being pushed away from the origin, the brushbot successfully returns to the origin again.

APPENDIX A
PROOF OF THEOREM 1

The line integral of $\frac{\partial B(x)}{\partial x}$ is path independent because it is the gradient of the scaler field B [53]. Let $x(t) := (1-t)x_n + tx_{n+1} = x_n + t(f_n(x_n) + g_n(x_n)u_n)$, where $t \in [0,1]$ parameterizes the line path between x_n and x_{n+1}, then $\frac{dB(x(t))}{dt} = \frac{\partial B(x(t))}{\partial x}(f_n(x_n) + g_n(x_n)u_n)$. Therefore, for any path A from x_n to x_{n+1}, it holds that

$$
\begin{aligned}
&B(x_{n+1}) - B(x_n) = \int_A \frac{\partial B(x)}{\partial x} \cdot \mathrm{d}x = \int_0^1 \frac{dB(x(t))}{dt}\mathrm{d}t \\
&\geq \int_0^1 \left(\frac{\partial B(x_n)}{\partial x} - \nu t (f_n(x_n) + g_n(x_n)u_n)^{\mathsf{T}} \right) \\
&(f_n(x_n) + g_n(x_n)u_n)\mathrm{d}t \\
&= \frac{\partial B(x_n)}{\partial x}(f_n(x_n) + g_n(x_n)u_n) \\
&- \frac{\nu}{2} \left\| f_n(x_n) + g_n(x_n)u_n \right\|^2_{\mathbb{R}^{n_x}}. \qquad \text{(A.1)}
\end{aligned}
$$

The inequality implies that $B(x_{n+1}) - B(x_n)$ is greater than or equal to that in the case when $\frac{\partial B(x)}{\partial x}$ decrease along the line path at the maximum rate. When (13) is satisfied, it holds from (A.1) that

$$
B(x_{n+1}) - B(x_n) \geq -\eta B(x_n),
$$

which is the discrete-time exponential control barrier certificate defined in (3). Equation (13) can be rewritten as

$$
\begin{aligned}
&\frac{\partial B(x_n)}{\partial x}(f_n(x_n) + g_n(x_n)u_n) \\
&- \frac{\nu}{2} \left\| f_n(x_n) + g_n(x_n)u_n \right\|^2_{\mathbb{R}^{n_x}} \geq -\eta B(x_n). \qquad \text{(A.2)}
\end{aligned}
$$

The first term in the left hand side of (A.2) is affine to u_n, the second term is the combination of a concave function $-\frac{\nu}{2}\|\cdot\|^2_{\mathbb{R}^{n_x}}$ and an affine function of u_n, which is concave. Therefore, the left hand side of (A.2) is a concave function, and the inequality (A.2) defines a convex constraint.

APPENDIX B
PROOF OF PROPOSITION 2

Since $\kappa(u,v) = \mathbf{1}(u) = 1, \forall u, v \in \mathcal{U}$ is a positive definite kernel, it defines the unique RKHS given by $\mathrm{span}\{\mathbf{1}\}$, which is complete because it is a finite-dimensional space. For any $\varphi := \alpha\mathbf{1} \in \mathcal{H}_c$, $\langle \varphi, \varphi \rangle_{\mathcal{H}_c} = \alpha^2 \geq 0$ and the equality holds if and only if $\alpha = 0$, or equivalently, $\varphi = 0$. The symmetry and the linearity also hold, and hence $\langle \cdot, \cdot \rangle_{\mathcal{H}_c}$ defines the inner product. For any $u \in \mathcal{U}$, it holds that $\langle \varphi, \kappa(\cdot, u) \rangle_{\mathcal{H}_u} = \langle \alpha\mathbf{1}, \mathbf{1} \rangle_{\mathcal{H}_u} = \alpha = \varphi(u)$. Therefore, the reproducing property is satisfied.

APPENDIX C
PROOF OF THEOREM 2

The following lemmas are used to prove the theorem.

Lemma 1 ([54, Theorem 2]). *Let $\mathcal{X} \subset \mathbb{R}^{n_x}$ be any set with nonempty interior. Then, the RKHS associated with the Gaussian kernel for an arbitrary scale parameter $\sigma > 0$ does not contain any polynomial on $\mathcal{X}$, including the nonzero constant function.*

Lemma 2. *Assume that $\mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathcal{U} \subset \mathbb{R}^{n_u}$ have nonempty interiors. Then, the intersection of the RKHS $\mathcal{H}_u$ associated with the kernel $\kappa(u,v) := u^{\mathsf{T}}v$, $u, v \in \mathcal{U}$, and the RKHS $\mathcal{H}_c := \{\varphi : \mathcal{U} \to \mathbb{R} | \exists \alpha \in \mathbb{R}, \varphi(u) = \alpha\}$ is $\{0\}$, i.e.*

$$
\mathcal{H}_c \cap \mathcal{H}_u = \{0\}.
$$

Proof. It is obvious that the function $\varphi(u) = 0, \forall u \in \mathcal{U}$ is an element of both of the RKHSs (vector spaces) $\mathcal{H}_u$ and $\mathcal{H}_c$. Therefore, it is enough to show that there exists $u \in \mathcal{U}$ satisfying that $\varphi(u) \neq \varphi(u^{\mathrm{int}})$, $u^{\mathrm{int}} \in \mathrm{int}(\mathcal{U})$, where $\mathrm{int}(\mathcal{U})$

denotes the interior of $\mathcal{U}$, for any $\varphi \in \mathcal{H}_u \setminus \{0\}$. Assume that $\varphi(v) \neq 0$ for some $v \in \mathcal{U}$. From [49, Theorem 3], the RKHS $\mathcal{H}_u$ is expressed as $\mathcal{H}_u = \mathrm{span}\{\kappa(\cdot, u)\}_{u\in\mathcal{U}}$, which is finite dimension, implying that any function in $\mathcal{H}_u$ is linear. Since there exists $u = u^{\mathrm{int}} + \varrho v \in \mathcal{U}$ for some $\varrho > 0$, it is proved that

$$\varphi(u) = \varphi(u^{\mathrm{int}} + \varrho v) = \varphi(u^{\mathrm{int}}) + \varrho\varphi(v) \neq \varphi(u^{\mathrm{int}}).$$

□

Lemma 3 ([55, Proposition 1.3]). *Given vector spaces $\mathcal{H}_1$ and $\mathcal{H}_2$. If $\mathcal{H}_2 = \mathcal{H}_{21} \oplus \mathcal{H}_{22}$, then*

$$\mathcal{H}_1 \otimes \mathcal{H}_{21} \cap \mathcal{H}_1 \otimes \mathcal{H}_{22} = \{0\},$$

i.e,

$$\mathcal{H}_1 \otimes \mathcal{H}_2 = (\mathcal{H}_1 \otimes \mathcal{H}_{21}) \oplus (\mathcal{H}_1 \otimes \mathcal{H}_{22}).$$

Lemma 4. *Given $\mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathcal{U} \subset \mathbb{R}^{n_u}$, let $\mathcal{H}_1$, $\mathcal{H}_2$, and $\mathcal{H}$ be associated with the Gaussian kernels $\kappa_1(x,y) := \frac{1}{(\sqrt{2\pi}\sigma)^{n_x}} \exp\left(-\frac{\|x-y\|^2_{\mathbb{R}^{n_x}}}{2\sigma^2}\right)$, $x, y \in \mathcal{X}$, $\kappa_2(u,v) := \frac{1}{(\sqrt{2\pi}\sigma)^{n_u}} \exp\left(-\frac{\|u-v\|^2_{\mathbb{R}^{n_u}}}{2\sigma^2}\right)$, $u, v \in \mathcal{U}$, and $\kappa([x^\mathsf{T}, u^\mathsf{T}]^\mathsf{T}, [y^\mathsf{T}, v^\mathsf{T}]^\mathsf{T}) := \frac{1}{(\sqrt{2\pi}\sigma)^{n_x+n_u}} \exp\left(-\frac{\left\|[x^\mathsf{T},u^\mathsf{T}]^\mathsf{T}-[y^\mathsf{T},v^\mathsf{T}]^\mathsf{T}\right\|^2_{\mathbb{R}^{n_x+n_u}}}{2\sigma^2}\right)$, $x, y \in \mathcal{X}$, $u, v \in \mathcal{U}$, respectively, for an arbitrary $\sigma > 0$. Then, by regarding a function in $\mathcal{H}_1 \otimes \mathcal{H}_2$ as a function over the input space $\mathcal{X} \times \mathcal{U} \subset \mathbb{R}^{n_x+n_u}$, it holds that*

$$\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2.$$

Proof. $\mathcal{H}_1 \otimes \mathcal{H}_2$ has the reproducing kernel defined by

$$\begin{aligned}
&\kappa_\otimes([x^\mathsf{T}, u^\mathsf{T}]^\mathsf{T}, [y^\mathsf{T}, v^\mathsf{T}]^\mathsf{T}) := \kappa_1(x,y)\kappa_2(u,v)\\
&= \frac{1}{(\sqrt{2\pi}\sigma)^{n_x}(\sqrt{2\pi}\sigma)^{n_u}}\\
&\exp\left(-\frac{\|x-y\|^2_{\mathbb{R}^{n_x}}}{2\sigma^2}\right)\exp\left(-\frac{\|u-v\|^2_{\mathbb{R}^{n_u}}}{2\sigma^2}\right)\\
&= \frac{1}{(\sqrt{2\pi}\sigma)^{n_x+n_u}} \exp\left(-\frac{\|x-y\|^2_{\mathbb{R}^{n_x}} + \|u-v\|^2_{\mathbb{R}^{n_u}}}{2\sigma^2}\right)\\
&= \kappa([x^\mathsf{T}, u^\mathsf{T}]^\mathsf{T}, [y^\mathsf{T}, v^\mathsf{T}]^\mathsf{T}).
\end{aligned}$$

This verifies the claim. □

We are now ready to prove Theorem 2.

Proof of Theorem 2. By Lemmas 2 and 3, it is derived that $\mathcal{H}_f \otimes \mathcal{H}_c \cap \mathcal{H}_g \otimes \mathcal{H}_u = \{0\}$. By Lemmas 1, 3, and 4, it holds that $\mathcal{H}_p \cap \mathcal{H}_f \otimes \mathcal{H}_c = \{0\}$ and $\mathcal{H}_p \cap \mathcal{H}_g \otimes \mathcal{H}_u = \{0\}$. □

Appendix D
Proof of Theorem 3

We show that the operator $U : \mathcal{H}_Q \to \mathcal{H}_\psi$, which maps $\varphi^Q \in \mathcal{H}_Q$ to a function $\varphi \in \mathcal{H}_\psi, \varphi([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) = \varphi^Q(z) - \gamma\varphi^Q(w)$ where $\gamma \in (0,1)$, $z, w \in \mathcal{Z}$, is bijective. First, for any $\varphi_1^Q, \varphi_2^Q \in \mathcal{H}_Q$,

$$\begin{aligned}
&U(\varphi_1^Q + \varphi_2^Q)([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T})\\
&= (\varphi_1^Q + \varphi_2^Q)(z) - \gamma(\varphi_1^Q + \varphi_2^Q)(w)\\
&= (\varphi_1^Q(z) - \gamma\varphi_1^Q(w)) + (\varphi_2^Q(z) - \gamma\varphi_2^Q(w))\\
&= U(\varphi_1^Q)([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) + U(\varphi_2^Q)([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}),\ \forall z, w \in \mathcal{Z},
\end{aligned}$$

and

$$\begin{aligned}
&U(\alpha\varphi_1^Q)([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T})\\
&= \alpha\varphi_1^Q(z) - \gamma\alpha\varphi_1^Q(w) = \alpha(\varphi_1^Q(z) - \gamma\varphi_1^Q(w))\\
&= \alpha U(\varphi_1^Q)([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}),\ \forall\alpha \in \mathbb{R},\ \forall z, w \in \mathcal{Z},
\end{aligned}$$

from which the linearity holds. Second, the mapping U is surjective by definition. Therefore, it is enough to show that $\ker(U) = 0$ [56]. This is shown in the following.

$$\begin{aligned}
&U(\varphi^Q)([z^\mathsf{T}, z^\mathsf{T}]^\mathsf{T}) = (1-\gamma)\varphi^Q(z) = 0,\\
&\forall z \in \mathcal{Z}, \forall\varphi^Q \in \ker(U) \Longrightarrow \varphi^Q = 0.
\end{aligned}$$

Therefore, the space $\mathcal{H}_\psi$ with the inner product defined in (22) is isometric to the RKHS H_Q, and hence is a Hilbert space. Next, we show that $\mathcal{H}_\psi$ is an RKHS. Because $\kappa^Q(\cdot, z) - \gamma\kappa^Q(\cdot, w) \in \mathcal{H}_Q$, it is true that $\kappa(\cdot, [z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}) \in \mathcal{H}_\psi$. Moreover, it holds that

$$\begin{aligned}
&\left\langle \kappa(\cdot, [z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}), \kappa(\cdot, [\tilde{z}^\mathsf{T}, \tilde{w}^\mathsf{T}]^\mathsf{T})\right\rangle_{\mathcal{H}_\psi}\\
&= \left\langle \kappa^Q(\cdot, z) - \gamma\kappa^Q(\cdot, w), \kappa^Q(\cdot, \tilde{z}) - \gamma\kappa^Q(\cdot, \tilde{w})\right\rangle_{\mathcal{H}_Q}\\
&= \left(\kappa^Q(z, \tilde{z}) - \gamma\kappa^Q(z, \tilde{w})\right) - \gamma\left(\kappa^Q(w, \tilde{z}) - \gamma\kappa^Q(w, \tilde{w})\right)\\
&= \kappa([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}, [\tilde{z}^\mathsf{T}, \tilde{w}^\mathsf{T}]^\mathsf{T}),
\end{aligned}$$

and that

$$\begin{aligned}
&\left\langle \varphi, \kappa(\cdot, [z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T})\right\rangle_{\mathcal{H}_\psi} = \left\langle \varphi^Q, \kappa^Q(\cdot, z) - \gamma\kappa^Q(\cdot, w)\right\rangle_{\mathcal{H}_Q}\\
&= \varphi^Q(z) - \gamma\varphi^Q(w) = \varphi([z^\mathsf{T}, w^\mathsf{T}]^\mathsf{T}),\ \forall\varphi \in \mathcal{H}_\psi.
\end{aligned}$$

Therefore, $\kappa(\cdot,\cdot) : \mathcal{Z}^2 \times \mathcal{Z}^2 \to \mathbb{R}$ is the reproducing kernel with which the RKHS $\mathcal{H}_\psi$ is associated.

Appendix E
Proof of Corollary 1

From the definition of the inner product in the RKHS $\mathcal{H}_\psi$, it follows that

$$\begin{aligned}
&\left\|Q_n^{\phi^{(n+1)}} - Q_n^{\phi^*}\right\|_{\mathcal{H}_Q} = \left\|\psi_n^{(n+1)} - \psi_n^*\right\|_{\mathcal{H}_\psi}\\
&\leq \left\|\psi_n^{(n)} - \psi_n^*\right\|_{\mathcal{H}_\psi} = \left\|Q_n^{\phi^{(n)}} - Q_n^{\phi^*}\right\|_{\mathcal{H}_Q}.
\end{aligned}$$

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 1998.

[2] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, vol. 9, no. 3, 2009.

[3] D. Liberzon, *Calculus of variations and optimal control theory: a concise introduction*. Princeton University Press, 2011.

[4] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Proc. NIPS*, 2017.

[5] F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause, "Safe learning of regions of attraction for uncertain, nonlinear systems with Gaussian processes," in *Proc. CDC*, 2016, pp. 4661–4666.

[6] J. Schreiter, D. Nguyen-Tuong, M. Eberts, B. Bischoff, H. Markert, and M. Toussaint, "Safe exploration for active learning with Gaussian processes," in *Proc. ECML PKDD*, 2015, pp. 133–149.

[7] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, "Reachability-based safe learning with Gaussian processes," in *Proc. CDC*, 2014, pp. 1424–1431.

[8] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, 2016.

[9] H. B. Ammar, R. Tutunov, and E. Eaton, "Safe policy search for lifelong reinforcement learning with sublinear regret," in *Proc. ICML*, 2015, pp. 2361–2369.

[10] D. A. Niekerk, B. V. and B. Rosman, "Online constrained model-based reinforcement learning," in *Proc. AUAI*, 2017.

[11] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. ICML*, 2017.

[12] P. Abbeel and A. Y. Ng, "Exploration and apprenticeship learning in reinforcement learning," in *Proc. ICML*, 2005, pp. 1–8.

[13] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," *arXiv preprint arXiv:1710.05472*, 2017.

[14] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 1437–1480, 2015.

[15] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[16] P. Geibel, "Reinforcement learning for MDPs with constraints," in *Proc. ECML*, vol. 4212, 2006, pp. 646–653.

[17] S. P. Coraluppi and S. I. Marcus, "Risk-sensitive and minimax control of discrete-time, finite-state Markov decision processes," *Automatica*, vol. 35, no. 2, pp. 301–309, 1999.

[18] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*. MIT press Cambridge, 2006, vol. 1.

[19] X. Xu, P. Tabuada, J. W. Grizzle, and A. D. Ames, "Robustness of control barrier functions for safety critical control," *Proc. IFAC*, vol. 48, no. 27, pp. 54–61, 2015.

[20] P. Wieland and F. Allgöwer, "Constructive safety using control barrier functions," *Proc. IFAC*, vol. 40, no. 12, pp. 462–467, 2007.

[21] P. Glotfelter, J. Cortés, and M. Egerstedt, "Nonsmooth barrier functions with applications to multi-robot systems," *IEEE Control Systems Letters*, vol. 1, no. 2, pp. 310–315, 2017.

[22] L. Wang, A. D. Ames, and M. Egerstedt, "Safety barrier certificates for collisions-free multirobot systems," *IEEE Trans. Robotics*, 2017.

[23] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs with application to automotive safety systems," *arXiv preprint arXiv:1609.06408*, 2016.

[24] A. Agrawal and K. Sreenath, "Discrete control barrier functions for safety-critical control of discrete systems with application to bipedal robot navigation," in *Proc. RSS*, 2017.

[25] W. Liu, J. Príncipe, and S. Haykin, *Kernel Adaptive Filtering*. New Jersey: Wiley, 2010.

[26] K. R. Müller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001.

[27] B. Schöelkopf and A. Smola, *Learning with kernels*. MIT Press, Cambridge, 2002.

[28] M. Yukawa, "Multikernel adaptive filtering," *IEEE Trans. Signal Processing*, vol. 60, no. 9, pp. 4672–4682, Sept. 2012.

[29] ——, "Adaptive learning in Cartesian product of reproducing kernel Hilbert spaces," *IEEE Trans. Signal Processing*, vol. 63, no. 22, pp. 6037–6048, Nov. 2015.

[30] O. Toda and M. Yukawa, "Online model-selection and learning for nonlinear estimation based on multikernel adaptive filtering," *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, vol. 100, no. 1, pp. 236–250, 2017.

[31] M. Ohnishi and M. Yukawa, "Online learning in L^2 space with multiple Gaussian kernels," in *Proc. EUSIPCO*, 2017, pp. 1594–1598.

[32] ——, "Online nonlinear estimation via iterative L^2-space projections: Reproducing kernel of subspace," *arXiv preprint arXiv:1712.04573*, 2017.

[33] L. Baird, "Residual algorithms: Reinforcement learning with function approximation," in *Proc. ICML*, 1995, pp. 30–37.

[34] S. Mahadevan, B. Liu, P. Thomas, W. Dabney, S. Giguere, N. Jacek, I. Gemp, and J. Liu, "Proximal reinforcement learning: A new theory of sequential decision making in primal-dual spaces," *arXiv preprint arXiv:1405.6757*, 2014.

[35] Y. Engel, S. Mannor, and R. Meir, "Reinforcement learning with Gaussian processes," in *Proc. ICML*, 2005, pp. 201–208.

[36] D. Ormoneit and Ś. Sen, "Kernel-based reinforcement learning," *Machine learning*, vol. 49, no. 2, pp. 161–178, 2002.

[37] X. Xu, D. Hu, and X. Lu, "Kernel-based least squares policy iteration for reinforcement learning," *IEEE Trans. Neural Networks*, vol. 18, no. 4, pp. 973–992, 2007.

[38] B. Bethke, J. P. How, and A. Ozdaglar, "Kernel-based reinforcement learning using Bellman residual elimination," in *MIT Working Paper*, 2008.

[39] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[40] G. Tesauro, "Temporal difference learning and TD-Gammon," *Communications of the ACM*, vol. 38, no. 3, pp. 58–68, 1995.

[41] J. N. Tsitsiklis and B. Van R., "Analysis of temporal-diffference learning with function approximation," in *Advances in Neural Information Processing Systems*, 1997, pp. 1075–1081.

[42] J. A. Boyan, "Least-squares temporal difference learning," in *Proc. ICML*, 1999, pp. 49–56.

[43] R. S. Sutton, H. R. Maei, and C. Szepesvári, "A convergent $O(n)$ temporal-difference algorithm for off-policy learning with linear function approximation," in *Advances in Neural Information Processing Systems*, 2009, pp. 1609–1616.

[44] M. Geist and O. Pietquin, "A brief survey of parametric value function approximation," *Rapport Interne, Supélec*, 2010.

[45] N. Aronszajn, "Theory of reproducing kernels," *Trans. Amer. Math. Soc.*, vol. 68, no. 3, pp. 337–404, May 1950.

[46] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," *J. Mach. Learn. Res.*, vol. 2, pp. 67–93, 2001.

[47] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *Proc. IEEE ICASSP*, 2010, pp. 3734–3737.

[48] I. Yamada and N. Ogura, "Adaptive projected subgradient method for asymptotic minimization of sequence of nonnegative convex functions," *Numerical Functional Analysis and Optimization*, vol. 25, no. 7&8, pp. 593–617, 2004.

[49] A. Berlinet and A. C. Thomas, *Reproducing kernel Hilbert spaces in probability and statistics*. Kluwer, 2004.

[50] Y. Engel, S. Mannor, and R. Meir, "The kernel recursive least-squares algorithm," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2275–2285, Aug. 2004.

[51] M. L. Puterman and S. L. Brumelle, "On the convergence of policy iteration in stationary dynamic programming," *Mathematics of Operations Research*, vol. 4, no. 1, pp. 60–69, 1979.

[52] D. Pickem, L. Wang, P. Glotfelter, Y. Diaz-Mercado, M. Mote, A. Ames, E. Feron, and M. Egerstedt, "Safe, remote-access swarm robotics research on the robotarium," *arXiv preprint arXiv:1604.00640*, 2016.

[53] L. V. Ahlfors, "Complex analysis: an introduction to the theory of analytic functions of one complex variable," *New York, London*, p. 177, 1953.

[54] H. Q. Minh, "Some properties of Gaussian reproducing kernel Hilbert spaces and their implications for function approximation and learning theory," *Constructive Approximation*, vol. 32, no. 2, pp. 307–338, 2010.

[55] R. A. Ryan, *Introduction to tensor products of Banach spaces*. Springer Science & Business Media, 2013.

[56] G. Strang, *Introduction to linear algebra*. Wellesley-Cambridge Press Wellesley, MA, 1993, vol. 3.